PGPUB-DOCUMENT-NUMBER: 20020091807

PGPUB-FILING-TYPE: new

DOCUMENT-IDENTIFIER: US 20020091807 A1

TITLE: Automatic firmware update of processor nodes

PUBLICATION-DATE: July 11, 2002

US-CL-CURRENT: 709/221, 713/1

APPL-NO: 09/ 755832

DATE FILED: January 5, 2001

---------- KWIC --------

Summary of Invention Paragraph - BSTX (7):

[0006] However, one difficulty with a nodal architecture is coordinating an upgrade to node firmware. For instance, in order to replace a node, the replacement node may have to be at the same firmware level as the other nodes in the system in order to function properly. However, the firmware level of a replacement part may be substantially different than the other system nodes, especially if the replacement node contains an older version of firmware than the component being replaced. For instance, the replacement

node may have been in-stock at a warehouse for a significant period of time, during which one or more firmware updates were made to the system. Currently, there are two solutions to maintaining code levels on processor assemblies. The first involves maintaining replacement nodes in the warehouse at current firmware levels. This solution is costly and time consuming because it involves turning the stock every time a firmware change is made. The second solution is to have a maintenance technician update the firmware when a node is replaced at the customer site. This solution is also problematic because of the cost of having the technician apply the update and the potential of human error when applying the update.

Summary of Invention Paragraph - BSTX (12):

[0010] Described implementations provide an automatic update to firmware at computing nodes in a distributed nodal architecture. The descried implementations reduce the need to ensure that warehouse stock is "turned" or updated every time a firmware update is released. Further, with the described implementations, there is no need for a technician to update the node firmware before using the replacement node. Still further, the implementations increase the likelihood that the same version of the firmware is running on all the

**nodes of the nodal system to avoid incompatibility problems between different
firmware levels.**

Brief Description of Drawings Paragraph - DRTX (5):

**[0014] FIG. 3 illustrates the logic implemented in the firmware to automatically update the firmware on a node in accordance with preferred
embodiments of the present invention; and**

Detail Description Paragraph - DETX (12):

**[0026] At block 250, once the querying node has compared its code level with
the queried node code signature, the node will check to see if all the nodes
20, 40, 60, 80 in the nodal system 100 have been checked. If all nodes have
not been checked, then the checking node queries (at block 210) the next node,
determined at block 260, for the code signature. This logic loop will be
performed until all the nodes 20, 40, 60, 80 have been queried. If a higher
code level is found at another node, then the Location Parameter is updated (at
block 240) with the location and firmware level of the node. If the same
higher code level is found in two or more nodes, then the logic of FIG. 2 will
retain the location of the first node with the higher code level. If (at block
270) a higher firmware level was found at another node, then (at block 280) the
Firmware Update Routine will initiate the copying of the higher**

code according
to logic described in FIG. 3. However, if no higher firmware 28a, b, c, d
exists at another node, then, at block 290, the Code Check Routine terminates.

Detail Description Paragraph - DETX (13):

[0027] FIG. 3 illustrates logic implemented in the firmware to automatically
update the nodes that determined that other nodes have a higher code level.
The processor 22a, b, c, or d then selects the node with the higher code level
indicated in the Location Parameter and requests a copy of the code image (at
block 310) from that node. The node receiving the request then broadcasts a
copy of its firmware to all the nodes which requested a copy of its firmware.
At block 320, all the requesting nodes receive a copy of the higher level
firmware from the identified node. At block 330, the Firmware Update routine
at each requesting node then rewrites the programmable memory with the copy of
the higher level code image received from the other node. At block 340, the
updated nodes reset themselves and the new firmware image is activated on those
nodes. The reset nodes will execute the code check routine again in FIG. 2 and
terminate upon determining that no higher firmware level exist in the nodal
system 100. Additionally, in the event of a code update failure, the code

update routine (FIG. 3) may perform a self node reset as a error recovery
action. For instance, if a failure occurs while doing the code update, then a
reset may restart the process of the code update.

Detail Description Paragraph - DETX (15):

[0029] The preferred embodiments are also applicable during a system wide
code update process described in FIG. 4. Typically, the system 100 can have
updated firmware delivered through a host system, debug port, or external
interface located on one or more nodes. At block 400, upon a signal from the
external interface, the code update routine is initiated at the local node.
Higher level code is then introduced (at block 410) to the local node through
the connection made with the host system, debug port, or other external
interface. The programmable memory on the local node is then updated (at block
420) to include the new version of the firmware. At block 430, the remaining
nodes are then reset through a remote reset or reboot message, electrical
signal or manual process, e.g., switching the power on and off, etc. Once the
nodes are reset, the nodes go through the same code verification process
described in FIG. 2 (at block 440). The automatic update routine of FIG. 3
will ensure that the code level will be the same at all the nodes.

	Type	L #	Hits	Search Text	DBs	Time Stamp
1	BRS	L1	11	firmware adj5 updat$3 with node	USPAT; US-PGPUB; EPO; JPO; IBM_TDB	2004/06/08 10:52